Taylor Diagrams

Â

Goal: Describe how to plot simple Taylor diagrams.

Before running the tutorial below, type *"python"* or *"cdat"* at the command line.Â You will see the python prompt appear (i.e., ">>>"). You can now enter the command lines below.

You can ***view***Â or ***download***Â the full source code. To run the source code at the command line, type: *"python td.py"*

The data must have the following shape: (npoints,2)
Where the first dimension length (npoints) represents the number of points to plot, and the second dimension has 2 value, the first one representing the "standard deviation" and the second one the "correlation"

First let's create a set of 3 points:
Point 1: standard deviation is 1.2, correlation .9
Point 2: standard deviation is 0.7, correlation .6
Point 3: standard deviation is 1.1, correlation .8

```
# MV - Masked Variable (MV): A masked array having a domain and metadata.
#      Computations carry along the domain and metadata information where
#      possible. A masked variable in memory is referred to as transient
#      variable and a masked variable in a dataset is called a file variable.
import MV
data = MV.array([[1.2,.9],[0.7,.6],[1.1,.8] ])
print data
```

```
[[ 1.2, 0.9,]
 [ 0.7, 0.6,]
 [ 1.1, 0.8,] ]
```

Now we will need to create a new taylor diagram methods using the vcs module

```
# vcs - Visualization and control System 1D and 2D plotting routines.
import vcs
x=vcs.init()
td=x.createtaylordiagram('new')
x.plot(data,td)
```

Now let's say we want to assign a different color for each Marker
Point 1: Red (color 242 on default colormap)
Point 2: Blue (244)
Point 3: Green (243)

First we can list the attributes of our graphic method:

```
td.list()
```

```
 ----------Taylordiagram (Gtd) member (attribute) listings ----------
graphic method = Gtd
name = new
detail = 75
max = None
quadrans = 1
skillValues = [0.10000000000000001, 0.20000000000000001, 0.29999999999999999,
   0.40000000000000002, 0.5, 0.59999999999999998, 0.69999999999999996,
   0.80000000000000004, 0.90000000000000002, 0.94999999999999996]
skillColor = grey
skillDrawLabels = y
skillCoefficient = [1.0, 1.0, 1.0]
referencevalue = 1.0
referencecolor = black
arrowlength = 0.05
arrowangle = 20.0
arrowbase = 0.75
xticlabels1 = *
xmtics1 = *
yticlabels1 = *
ymtics1 = *
cticlabels1 = *
cmtics1 = *
Marker
    status =  ['on', 'on', 'on']
    line =  [None, None, None]
    id =  ['', '', '']
    id_size =  [30, 30, 30]
    id_color =  ['black', 'black', 'black']
    id_font =  [1, 1, 1]
    symbol =  ['dot', 'dot', 'dot']
    color =  ['black', 'black', 'black']
    size =  [5, 5, 5]
    xoffset =  [0.0, 0.0, 0.0]
    yoffset =  [0.0, 0.0, 0.0]
    line_color =  ['black', 'black', 'black']
    line_size =  [1.0, 1.0, 1.0]
    line_type =  ['solid', 'solid', 'solid']
```

The Marker part describe markers attributes, we can therefore set these as follow

```
# Note that here you can use either the color number
# or a "name"
td.Marker.color = ['red',244,'green']
x.clear()
x.plot(data,td)
```

Similarly the symbol could be changed and an "id" could be added

```
td.Marker.id_color = ['red',244,'green']
td.Marker.id = ['Point 1','Point 2','Point 3']
td.Marker.symbol=['dot','cross','circle']
x.clear()
x.plot(data,td)
```

Finally the reference (inner) circle and outer circle are controled as follow:

```
td.referencevalue = 2.
td.max = 2.5
x.clear()
x.plot(data,td)
```

Â